

# Addressing Overfitting in Deep Neural Networks

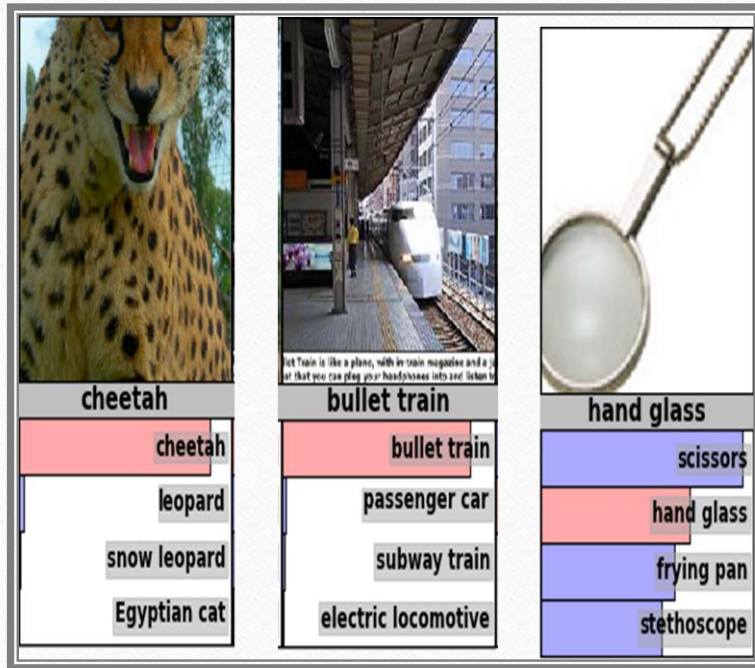
פתרון בעיית התאמת יתר ברשתות נוירונים עמוקות

---

& Brain-Inspired Method of Pre-processing a Deep Neural Network

טכניקה בהשראת המוח לעיבוד מקדים של רשתות נוירונים

## למה רשתות נוירונים?



- בעיות שלא ניתן לפתור בתכנות מפורש.
- הגישה היא להציג בשלב האימון labelled data. הרשת לומדת את היחס בין הקלט לפלט - כך שבשלב המבחן הרשת תוכל לתת חיזוי עבור מידע חדש.
- כיום למידה ברשתות עמוקות הן שיא הטכנולוגיה בבינה מלאכותית ומשווית להמצאת החשמל.



# רשתות נוירונים מלאכותיות בהשראת רשתות נוירונים מוחיות

- מאפיינים של נוירון ביולוגי:

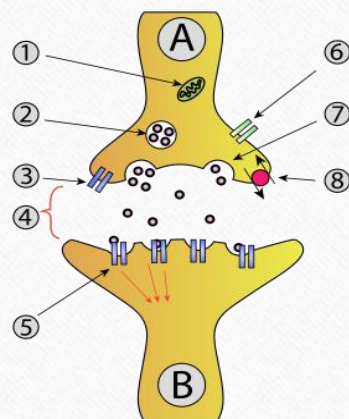
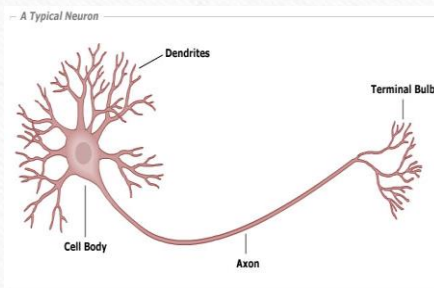
- חישוב בגוף התא האם יתרחש פוטנציאל פעולה

- all or none principle

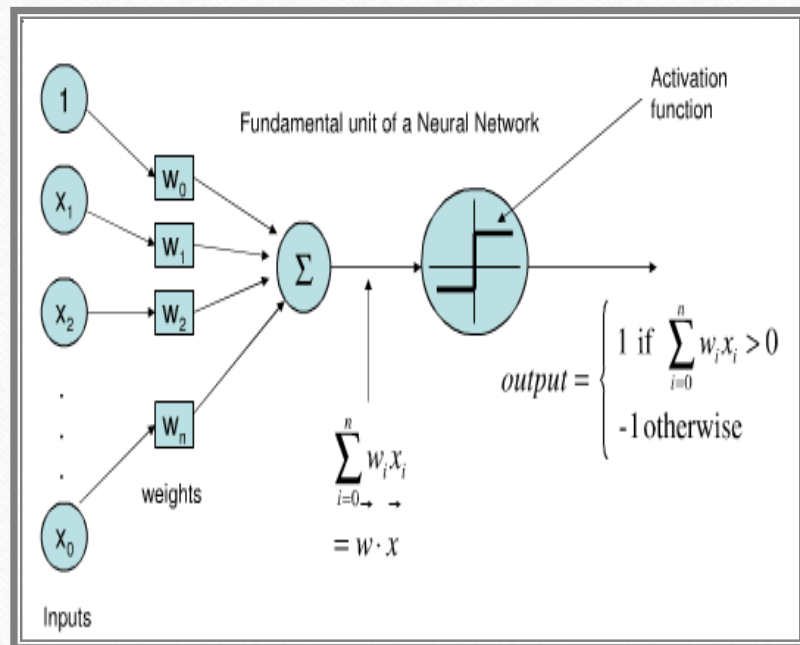
- הולכה של הפוטנציאל עד לסינפסה

- מעבר אינפורמציה בסינפסה

- למידה מבוטאת במוח באמצעות שינוי בחוזק הסינפטי



# McCulloch & Pitts Neuron and Perceptron



- בהתבסס על המאפיינים שהוזכרו בשקף הקודם פותח הנוירון המלאכותי ורשת הנוירונים המלאכותיות
- פונקציית אקטיבציה המקבילה לחישוב בגוף התא
- המשקולות הן המקבילה לסינפסה
- הלמידה מבוטאת באמצעות שינוי המשקולות כך שיתאימו ליחס בין הקלט לפלט בדוגמאות האימון
- נקודה למחשבה. איזה מאפיין מוחי עקרוני חסר ברשתות מלאכותיות?



# Cost Function & Gradient Descent

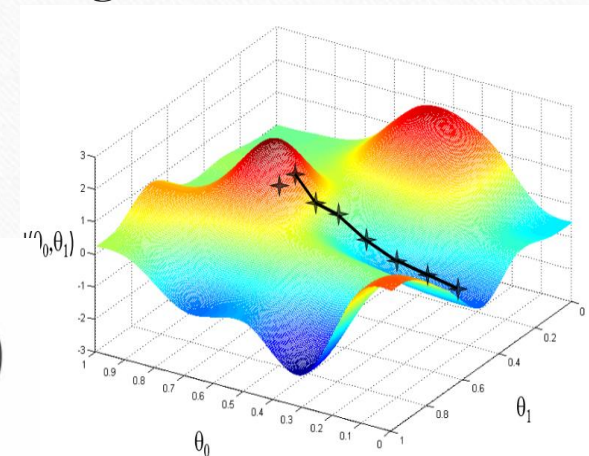
- פונקציית מחיר – מדד לשגיאה הכוללת של הרשת
- gradient descent – שינוי בערכי המשקולות כך שערך פונקציית המחיר תהיה מינימלית

$$E = \frac{1}{2} \sum_{n \in \text{training}} (t^n - y^n)^2$$

$$J(w, b) = \frac{1}{m} \sum_{i=1}^m L(\hat{y}^{(i)}, y^{(i)}) = -\frac{1}{m} \sum_{i=1}^m [(y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}))]$$

$$\Delta w_{ji} = \alpha(t_j - y_j)g'(h_j)x_i$$

repeat until convergence {  
 $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$  (for  $j = 0$  and  $j = 1$ )  
}

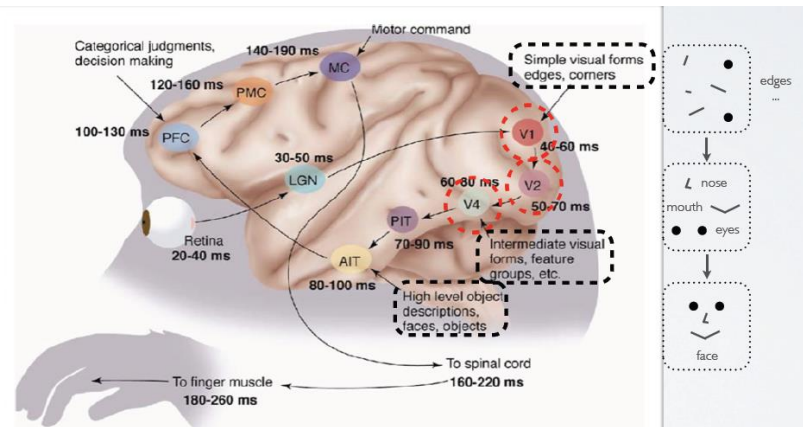


## מגבלות של רשת חז שכבתית

- בשנת 1957 כאשר פרנק רוזנבלט הציג את הפרספטרון הוא טען שמחשב מבוסס על הפרספטרון יוכל ללכת, לדבר, לראות, לשכפל את עצמו ולהיות בעל מודעות....
- רשת חז שכבתית מוגבלת לבעיות שניתנות להפרדה ליניארית
- איך מתגברים על מגבלות רשת חז שכבתית:
- Kernel טריק מתמטי, שינוי מאפייני הקלט כך שהבעיה תהיה ניתנת להפרדה ליניארית - לא עובד בבעיות מורכבות. רשת חז שכבתית לא אקספרסיבית מספיק.
- **רשת רב שכבתית עמוקה**



## רשת רב שכבתית



- הפתרון שהוצע בשנות ה 80 רשת רב שכבתית גם הוא בהשראת המוח.
- זיהו פנים במוח. שכבה ראשונה V1 מגיבה לצורות פשוטות, קצוות, וקשתות. שכבה שנייה, V2 מגיבה לצרופים של צורות השכבה הראשונה, וכך הלאה עד שבשכבה האחרונה יש ייצוגים גבוהים של אובייקטים מורכבים, כדוגמת זיהוי פנים וכו'

# אלגוריתם הלמידה ברשת רב שכבתית

## Backpropagation

- ברשת חד שכבתית המורכבת משכבת קלט ושכבת פלט אנו יודעים כיצד הנוירונים בשכבת הפלט צריכים להגיב.
- ברשת רב שכבתית לא ניתן לדעת כיצד הנוירונים בשכבות החבויים צריכים להגיב.

אתחל את המשקולות לערך רנדומלי קטן.  
הצג לרשת את תבנית הקלט וחשב את הפלט של הרשת.  
לכל נוירון  $k$  בשכבת הפלט חשב את גורם התיקון בצורה הבאה:

$$\delta_k = g'(\sum_i W_{ki} y_i)(t_k - o_k)$$

$\delta_k$  - גורם התיקון של נוירון הפלט  $k$   
 $g'$  - נגזרת של פונקציית האקטיבציה  
 $t_k$  - הפלט הרצוי  
 $o_k$  - הפלט של הרשת

$\sum_i W_{ki} y_i$  - סכום המכפלה של המשקולות שנכנסים לנוירון פלט  $k$ , עם הפלט של הנוירון המתאים למשקולת.

חשב את גורם השגיאה לכל נוירון בשכבה החבויה בצורה הבאה:

$$\delta_i = g'(\sum_j W_{ij} x_j) \sum_k W_{ik} \delta_k$$

$\delta_i$  - גורם התיקון של נוירון  $i$   
 $g'$  - נגזרת של פונקציית האקטיבציה  
 $\sum_j W_{ij} x_j$  - סכום המכפלה של המשקולות שנכנסים לנוירון החבוי  $i$ , עם הפלט של הנוירון המתאים שעושה סינפסה עם נוירון חבוי  $i$ .  
 $\sum_k W_{ik} \delta_k$  - סכום מכפלת המשקולות שיוצאות מנוירון  $i$ , עם גורם השגיאה.

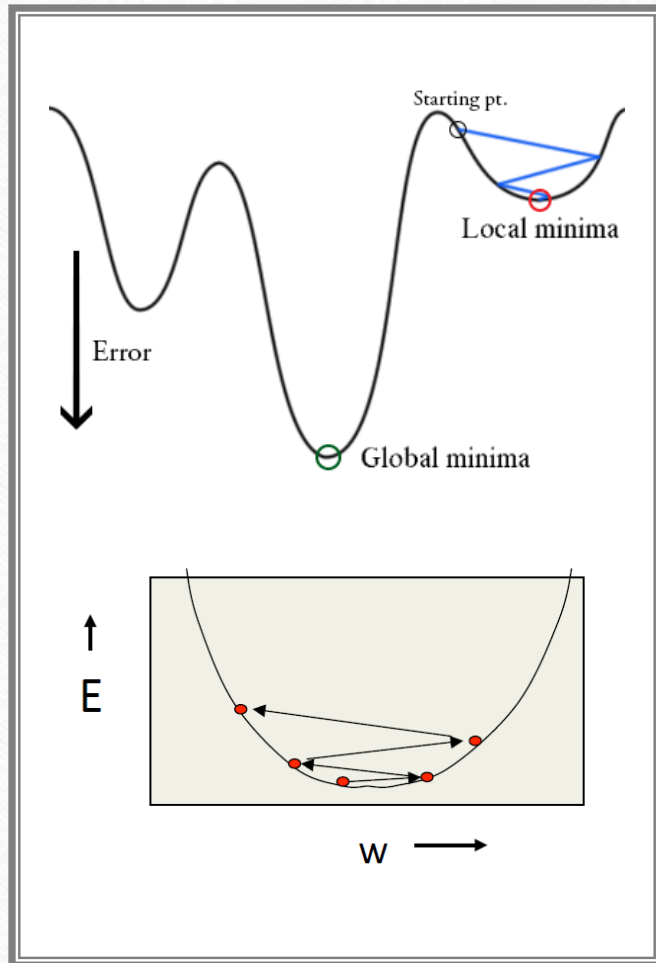
תקן כל משקולת עפ"י גורם השגיאה בצורה הבאה, הוסף לכל משקולת את הערך הבא:

$$\Delta W_{ki} = \eta \delta_k y_i$$

$\Delta W_{ki}$  - הערך שיש להוסיף למשקולת  
 $\eta$  - קבוע הלמידה  
 $\delta_k$  - גורם התיקון שחושב  
 $y_i$  - ערך הסיגנל של הנוירון

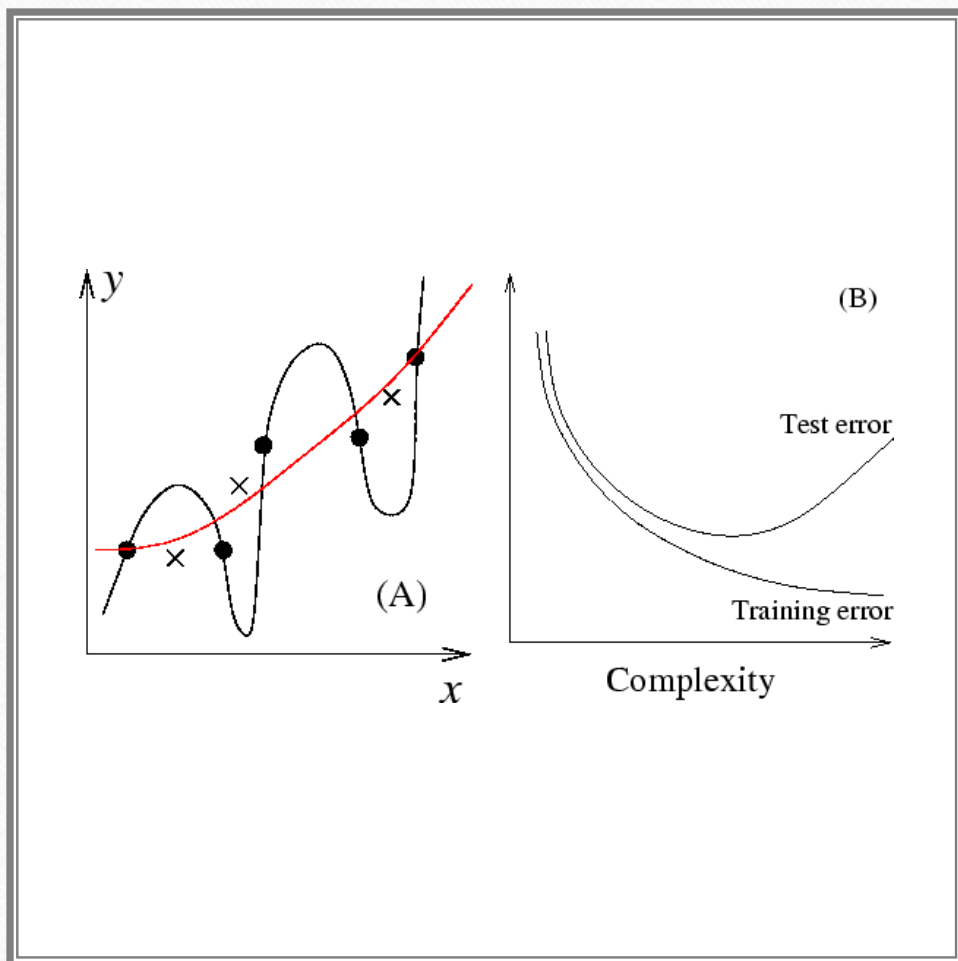


## בעיות ברשת רב שכבתית



- מינימום לוקלי – נקודות אוסף פתרונות:
- התאמת קבוע למידה (קבוע למידה גבוה אין התכנסות למינימום)
- מומנטום
- התאמת יתר - בעיה אינהרנטית לרשתות עמוקות מרובות שכבות ופרמטרים

## התאמת יתר ברשתות עמוקות



- הרשת מתאימה באופן מושלם לדוגמאות האימון אך ללא יכולת הכללה לדוגמאות חדשות.
- נובע ישירות ממורכבות הרשת ומריבוי השכבות והמשקולות.
- רשת עם שכבות חבויות רבות למעשה יכולה לשנן (Memorize) את היחס בין הקלט לפלט לכל דוגמא בדוגמאות האימון, ללא יכולת הכללה.



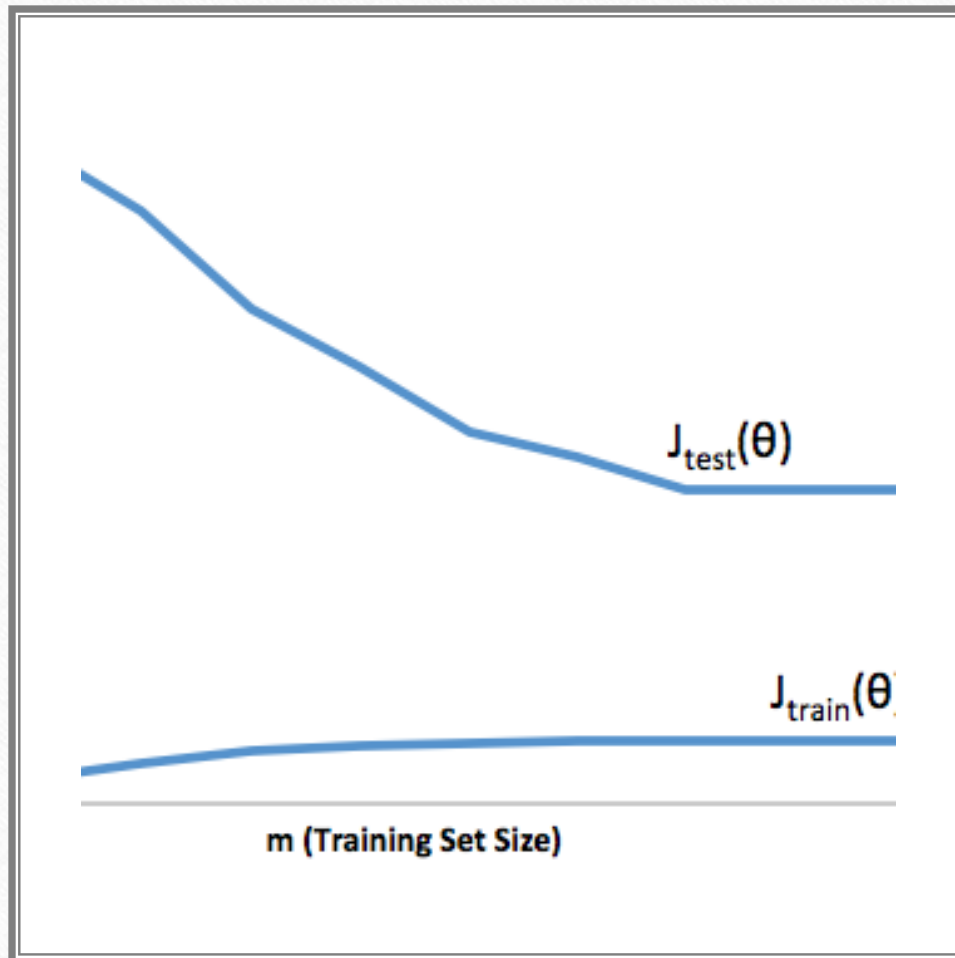
## בעיית התאמת יתר

---

- בעיית התאמת היתר מנעה מרשתות נוירונים עמוקות להניב תוצאות פרקטיות במשך יותר מ-20 שנה! כיום למידה עמוקה היא שיא הטכנולוגיה ומושווית להמצאת החשמל!
- לא נמצא פתרון טוב לבעיה משנות ה-80 עד לשנת 2006.
- למרות שבשנים אלו (1980-2006) פותחו מספר טכניקות להפחתת התאמת יתר

Weight decay, Weight sharing, Early stopping, Noise injection & Model averaging

## התאמת יתר שאלה מרכזית



- למה רק בשנת 2006 לפתע חוקרים התחילו לפתח טכניקות לפתרון הבעיה עד שכיום בשנת 2018 רשתות עמוקות הן שיא הטכנולוגיה?

- אלגוריתם הלמידה ברשתות עמוקות backprop פותח בשנות ה 80

- תשובה:**

1. מהירות ויעילות המעבדים

2. זמינות של דוגמאות אימון רבות ברשת האינטרנט (גורם מרכזי מעבר לכל הטכניקות שמפחית התאמת יתר) לרשת יש הזדמנות ללמוד את היחס האמתי בין הקלט לפלט



# Unsupervised Pretraining

טכניקה ראשונה שהחיתה את רשתות נוירונים עמוקות

שנת 2006-2007

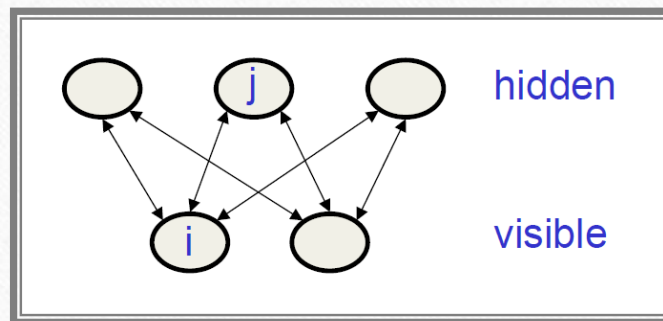
---

- בשנת 1985 ג'פרי הינטון המציא אלגוריתם וארכיטקטורה ל Unsupervised learning הקרויה Boltzmann Machine (BM). זהו למעשה פיתוח של Hopfield Network. האנרגיה הכללית של BM זהה לאנרגיה הכללית של Hopfield Network.
- ב Hopfield Network שואפים להגיע למינימום אנרגיה, וב BM למצב שנקרא Thermal equilibrium. האלגוריתם של BM אינו יעיל  $O(k^n)$ , ואינו שימושי.

# Contrastive Divergence

## Restricted Boltzmann Machine

- לקראת 2006, הינטון וצוותו, מצאו קיצור דרך לאלגוריתם המקורי, שאפשר לבצע Generative unsupervised learning ב-RBM באופן מהיר. לאלגוריתם הזה קוראים Contrastive divergence.
- האימון המסורתי של RBM ועדכון המשקולות הצריך מעבר סדרתי ארוך מאוד, בין היחידות הנראות (visible units) והיחידות החבויות. בשנת 2006 נמצא שמעבר אחד בין היחידות הנראות והיחידות החבויות ועדכון המשקולות על פי מעבר זה מייצר מודל טוב מספיק של הנתונים.





## RBM - Greedy Wise Layer Learning

---

- בנוסף, בשנת 2006 הינטון וצוותו, מצאו דרך לחבר RBM אחד על גבי השני ולאמן את כל ה RMBs בצורה מהירה (כך שמתקבל מודל עמוק אחד).
- האימון המהיר של RMBs אחד על השני מתבצע באמצעות Greedy wise layer learning, אימון של שכבה אחת אחרת בכל פעם.
- הם קראו למבנה של RBM אחד על השני Deep Belief Net (DBN).

Table 1: Error rates of Various Learning Algorithms on the MNIST Digit Recognition Task.

Version of MNIST Task	Learning Algorithm	Test Error
Permutation invariant	Our generative model: 784 → 500 → 500 ↔ 2000 ↔ 10	1.2
Permutation invariant	Support vector machine: degree 9 polynomial kernel	1.4
Permutation invariant	Backprop: 784 → 500 → 300 → 10 cross-entropy and weight-decay	1.5
Permutation invariant	Backprop: 784 → 800 → 10 cross-entropy and early stopping	1.5
Permutation invariant	Backprop: 784 → 500 → 150 → 10 squared error and on-line updates	2.9
Permutation invariant	Nearest neighbor: all 60,000 examples and L3 norm	2.8
Permutation invariant	Nearest neighbor: all 60,000 examples and L2 norm	3.1
Permutation invariant	Nearest neighbor: 20,000 examples and L3 norm	4.0
Permutation invariant	Nearest neighbor: 20,000 examples and L2 norm	4.4
Unpermuted images; extra data from elastic deformations	Backprop: cross-entropy and early-stopping convolutional neural net	0.4
Unpermuted de-skewed images; extra data from 2 pixel translations	Virtual SVM: degree 9 polynomial kernel	0.5
Unpermuted images	Shape-context features: hand-coded matching	0.6
Unpermuted images; extra data from affine transformations	Backprop in LeNet5: convolutional neural net	0.8
Unpermuted images	Backprop in LeNet5: convolutional neural net	0.9

## ניתוח השוואתי Generative Learning מול טכניקת אחרות

- בסיס הנתונים MNIST. ספרות שנכתבו בכתב יד. בסיס הנתונים מכיל 60,000 דוגמאות אימון ו 10,000 דוגמאות מבחן.
- Generative learning 1.25 אחוז טעות. 3 שכבות חביות.
- 2 שכבות חביות Backpropagation עם Weight decay 1.51 אחוז טעות.
- 2 שכבות חביות עם Backpropagation ועצירה מוקדמת. 1.53 אחוז טעות.



# ניתוח השוואתי עם ובלי Unsupervised Pretraining

pre-trained network	backprop training set size	train epochs	train cost per 100	train errs	valid. cost per 100	valid. errs	test cost per 100	test errs
Neta	50,000	33	0.12	1	6.49	129	6.22	122
Netb	50,000	56	0.04	0	7.81	118	6.21	116
Netc	50,000	63	0.03	0	8.12	118	6.73	124
Combined							5.75	110
Neta	60,000	33+16	<0.12	1			5.81	113
Netb	60,000	56+28	<0.04	0			5.90	106
Netc	60,000	63+31	<0.03	0			5.93	118
Combined							5.40	106
not pre-trained	60,000	119	<0.063	0			18.43	227

Table 1: Neta, Netb, and Netc were greedily pretrained on different, unlabeled, subsets of the training data that were obtained by removing disjoint validation sets of 10,000 images. After pretraining, they were trained on those same subsets using backpropagation. Then the training was continued on the full training set until the cross-entropy error reached the criterion explained in the text.

- Unsupervised pre-training ולאחר מכן  
1.1 Backpropagation אחוז טעות (ניתן להשוות עם  
שקף קודם)
- רשתות עם Backpropagation ללא הליך מקדים 2.27  
אחוז טעות.
- רשתות עם Unsupervised pre-training ולאחריו  
Backpropagation מניבות את התוצאות הטובות ביותר  
בשנת 2007.

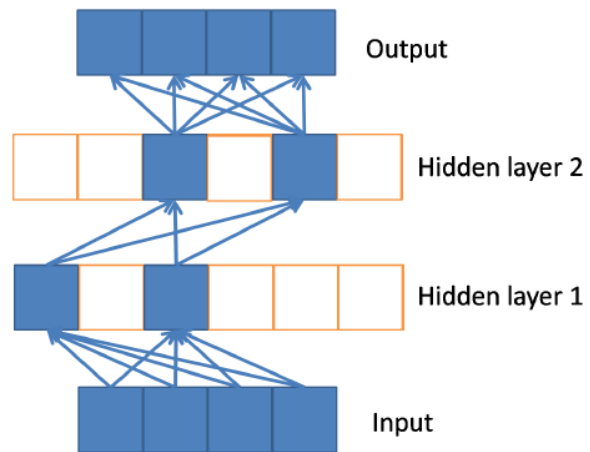
## מדוע Unsupervised Pre-Training מפחית התאמת יתר?

---

- **מנקודת ראות של אופטימיזציה.**
- נקודת התחלה טובה. הגרדיאנטים הראשוניים לפני התחלת Backprop כבר טובים ויש לבצע רק חיפוש לוקלי מנקודת התחלה טובה.
- **מנקודת ראות של התאמת יתר.**
- הקלט לבדו מכיל הרבה מאוד מידע, יותר מה Label.
- אנו משתמשים במידע המאוד יקר ש ה Label מספק לנו רק בסוף ב Fine tuning.
- ערכי המשקולות הסופיים נקבעים גם מהמידע שהקלט עצמו מכיל ללא ה Label

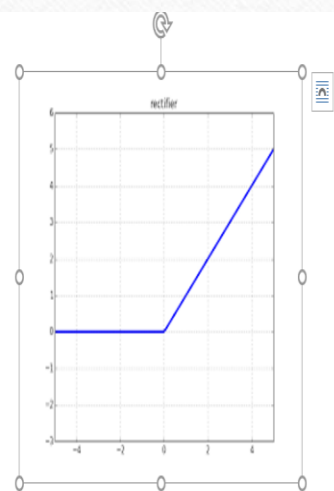


# Unsupervised Pretraining & Sparsity?



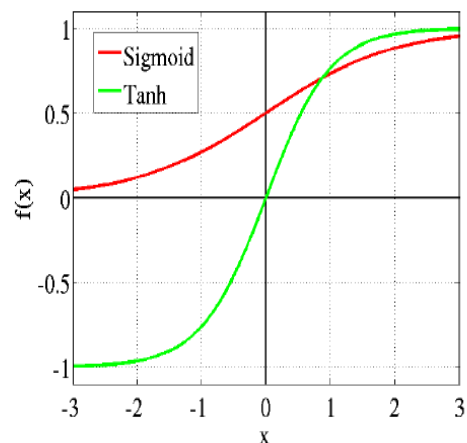
- נקודה תיאורטית מעניינת שלא נדונה בהרחבה בספרות המקצועית בהקשר להשפעה של Unsupervised pre-training, היא ש Unsupervised pre-training גורם למודל דליל
- Li, Luo, Yang, and Yuan(2013) מצאו ש Unsupervised pre-training גורם באופן ישיר ל Sparseness בשכבות החבויים. בנוסף, ככל שמבצעים Unsupervised pre-training ארוך יותר, כך מתקבל מודל דליל יותר.

# טכניקה שנייה: Deep Sparse Rectifier Neural Networks שנת 2011



$$f(x)=x \quad x>0$$

$$f(x)=0 \quad x\leq 0$$



- עד לשנת 2011 פונקציות האקטיבציה השכיחות היו Logistic sigmoid או Tanh
- Bengio (2011) ושות' מצאו 2 ממצאים עיקריים:

1. רשתות עמוקות עם Rectifier מניבות מדדי דיוק טובים יותר מרשתות עם פונקציות אקטיבציה שכיחות שעברו Unsupervised pre-training כלומר, Rectifier מפחית התאמת יתר טוב יותר מ Unsupervised pre-training לא Unsupervised pre-training משפרת את התאמת יתר ברשתות עם Rectifier מעבר לשיפור שה Rectifier גורם.
- 2.



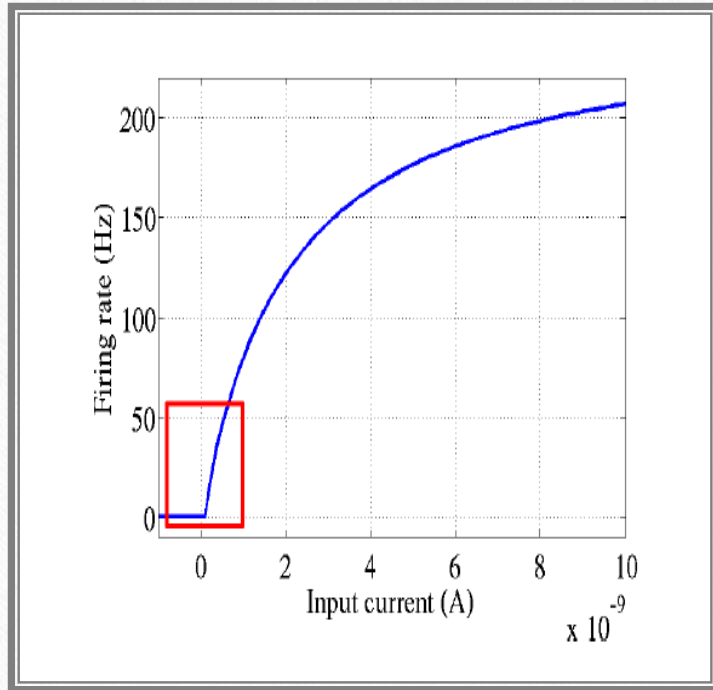
## השוואה בין Rectifier Neural Network לשיטות אחרות

Table 1: Test error on networks of depth 3. Bold results represent statistical equivalence between similar experiments, with and without pre-training, under the null hypothesis of the pairwise test with  $p = 0.05$ .

Neuron	MNIST	CIFAR10	NISTP	NORB
<i>With unsupervised pre-training</i>				
Rectifier	<b>1.20%</b>	<b>49.96%</b>	<b>32.86%</b>	<b>16.46%</b>
Tanh	<b>1.16%</b>	<b>50.79%</b>	<b>35.89%</b>	<b>17.66%</b>
<i>Without unsupervised pre-training</i>				
Rectifier	<b>1.43%</b>	<b>50.86%</b>	<b>32.64%</b>	<b>16.40%</b>
Tanh	<b>1.57%</b>	<b>52.62%</b>	<b>36.46%</b>	<b>19.29%</b>

- באופן גורף Rectifier טוב יותר מרשתות עם פונקציית אקטיבציה Tanh
  - Unsupervised pre-training מועיל רק לפונקציית אקטיבציה Tanh ולא ל Rectifier
  - Rectifier ללא Unsupervised pre-training, טוב יותר מרשתות אחרות עם Unsupervised pre-training
- מאמר זה היווה נקודת מפנה במחקר שעוסק ברשתות נוירונים והתאמת יתר. אין צורך יותר ב Unsupervised pre-training, אותו מרכיב שגרם לתחייה מחודשת של רשתות נוירונים עמוקות ב2006, דעך ב2011, לטובת Rectifier neural networks.

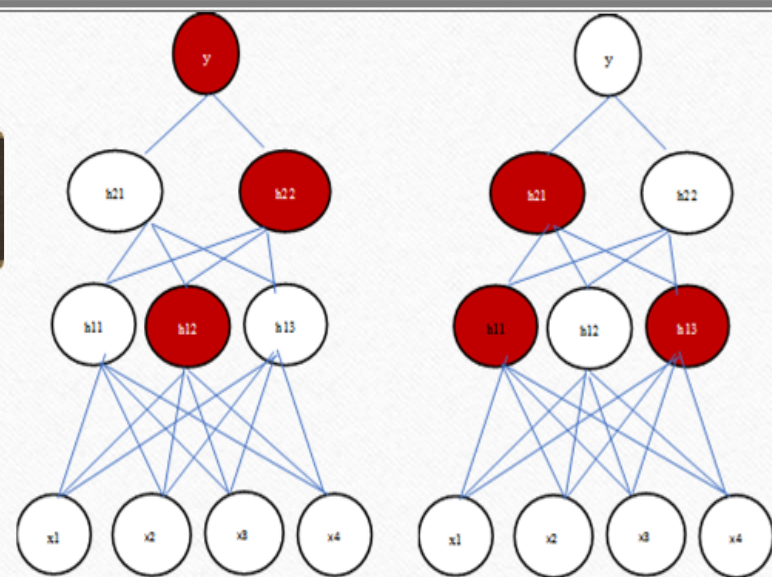
## מדוע ה Rectifier יעילה בהפחתת התאמת היתר?



- התאמה טובה יותר לפעילות של נוירון ביולוגי ו sparsity.
- הייתה מחשבה שהפונקציה הסיגמואידית ודומותיה מתאימות לתיאור פעילות של נוירון ביולוגי בגלל הרציפות וההגעה לרוויה.
- אך ב Zoom In הפונקציה הסיגמואידית מפספסת תכונה מאוד חשובה של הנוירון הביולוגי והיא Sparsity.
- **Sparse Coding אלמנט מרכזי בתגובה המוחית!**
- Rectifier מתאים לפעילות הנוירון הביולוגי.



# Sparse Coding



יתרון של Sparsity. לרשת מוצגת 2 דוגמאות קלט שונות (איור שמאלי ואיור ימני). עבור כל קלט רק תת קבוצה ספציפית מתוך סך כל הנוירונים פעילה כתגובה לקלט ספציפי. ניתן לבצע קלסיפיקציה מדויקת רק מלדעת איזו תת קבוצה ספציפית פעילה.

- בהינתן אוכלוסייה של נוירונים המטפלת בגירויים מסוג מסוים.
- רק תת קבוצה ספציפית של נוירונים מסך כל הנוירונים תגיב עבור כל גירוי ספציפי. מספיק לדעת איזה נוירונים פעילים ואיזה מושתקים פר גירוי.
- וזו הסיבה שהמוח הביולוגי יעיל מאוד בביצוע קלסיפיקציות ואבחנה בין גירויים דומים מאותו סוג

# Dropout 2012

SRIVASTAVA, HINTON, KRIZHEVSKY, SUTSKEVER AND SALAKHUTDINOV

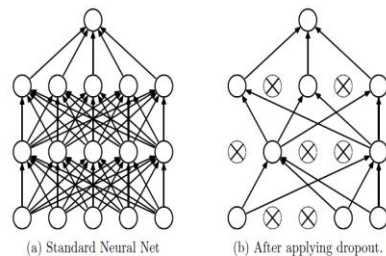


Figure 1: Dropout Neural Net Model. Left: A standard neural net with 2 hidden layers. Right: An example of a thinned net produced by applying dropout to the network on the left. Crossed units have been dropped.

DROPOUT

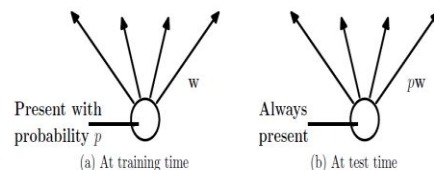


Figure 2: Left: A unit at training time that is present with probability  $p$  and is connected to units in the next layer with weights  $w$ . Right: At test time, the unit is always present and the weights are multiplied by  $p$ . The output at test time is same as the expected output at training time.

- בכל שלב מוחקים באופן זמני כל נוירון בהסתברות 0.5 (ניתן גם הסתברות אחרת).
- מריצים Backprop ומעדכנים את הפרמטרים.
- מחזירים את הנוירונים שהוסרו וחוזרים שוב על התהליך.
- בשלב המבחן: משתמשים בכל הנוירונים ובכל המשקולות של הרשת אחרי שמכפילים כל משקולת ב-0.5 בהסתברות להישארות.
- הרשת בזמן המבחן מחשבת בדיוק את הממוצע הגיאומטרי של החיזויים מכל ה- $2^H$  מודלים.



# השוואה Dropout vs Without Dropout

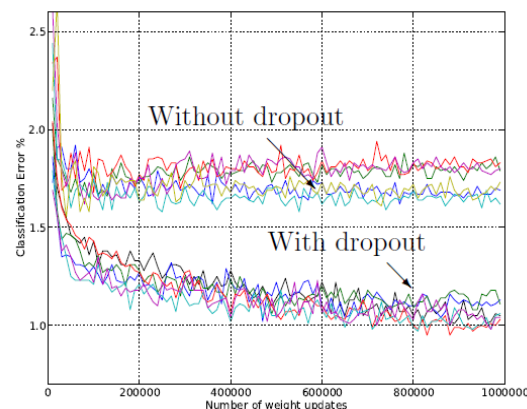


Figure 4: Test error for different architectures with and without dropout. The networks have 2 to 4 hidden layers each with 1024 to 2048 units.

6.1.1 MNIST

Method	Unit Type	Architecture	Error %
Standard Neural Net (Simard et al., 2003)	Logistic	2 layers, 800 units	1.60
SVM Gaussian kernel	NA	NA	1.40
Dropout NN	Logistic	3 layers, 1024 units	1.35
Dropout NN	ReLU	3 layers, 1024 units	1.25
Dropout NN + max-norm constraint	ReLU	3 layers, 1024 units	1.06
Dropout NN + max-norm constraint	ReLU	3 layers, 2048 units	1.04
Dropout NN + max-norm constraint	ReLU	2 layers, 4096 units	1.01
Dropout NN + max-norm constraint	ReLU	2 layers, 8192 units	0.95
Dropout NN + max-norm constraint (Goodfellow et al., 2013)	Maxout	2 layers, (5 × 240) units	0.94
DBN + finetuning (Hinton and Salakhutdinov, 2006)	Logistic	500-500-2000	1.18
DBM + finetuning (Salakhutdinov and Hinton, 2009)	Logistic	500-500-2000	0.96
DBN + dropout finetuning	Logistic	500-500-2000	0.92
DBM + dropout finetuning	Logistic	500-500-2000	0.79

Table 2: Comparison of different models on MNIST.

- על בסיס נתונים MNIST.
- ללא Dropout הרשתות מניבות מעל ל 1.5% כאשר רשתות עם Dropout מתקרבות ל 1 % טעות.
- Dropout משפרת את ביצועי הרשת גם בשילוב עם Rectifier וגם בשילוב עם Unsupervised pre-training
- רשת שעברה Unsupervised pre-training ביחד עם Dropout מניבה 0.79% טעות.

## מדוע Dropout מפחית התאמת יתר ברשתות עמוקות?

- **Model averaging**. שיטה ידועה עשרות שנים לפני Dropout, שבה מבצעים Mixture of experts. ב Dropout כל פעם אנו דוגמים רשת שונה מתוך  $2^H$  רשתות (ארכיטקטורות) אפשריות. מניבים חיזוי בהתאם לממוצע שהיה מתקבל מכל ה  $2^H$  רשתות.
- **מניעת Co-adaptation** בין נוירונים. במהלך ה Backprop הפרמטר לומד איך להשתנות כך שפונקציית המחיר תפחת בהינתן ערכי המשקולות האחרים שייתכן והם ערכים שגויים.
- ב Dropout כל יחידה חייבת להיות עצמאית, ולא יכולה להסתמך על ערכי משקולו אחרים, כי כל פעם היחידה צריכה לתפקד בארכיטקטורה שונה עם משקולות שונות.



# Dropout & Sparsity?

---

- עם Dropout רוב הנורונים היו מושתקים, פעילות 0, עבור רוב דוגמאות האימון. בעוד שללא Dropout לרוב הנורונים פעילות גבוהה עבור רוב דוגמאות האימון.

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1), 1929-1958

## סיכום קצר ומסקנה

---

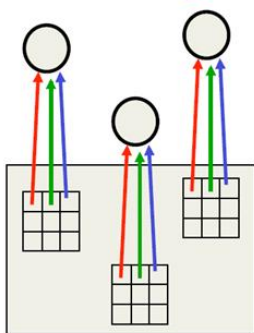
- סקרנו את שלושת השיטות המרכזיות שקידמו את רשתות עמוקות להיות שיא הטכנולוגיה:
- Unsupervised pre-training, Rectifier Neural Networks, & dropout

*Sparse hidden units activation* כאפקט מרכזי של שיטות אלו



# ImageNet Large Scale Visual Recognition Challenge

The red connections all have the same weight.



- תחרות משנת 2010.
- ImageNet בסיס נתונים. 15 מיליון תמונות ברזולוציה גבוהה, כאשר כולן מתויגות (Labeled). התמונות שייכות ל 20,000 קטגוריות.
- מאפשר התבוננות בהתפתחות של רשתות עמוקות.
- הרשתות למשימות זיהוי ויזואלי הן מסוג ConvNet

# סקירה מהירה של התפתחות הרשתות וביצועיהן ברשתות אלו יישמו את השיטות שתוארו בעבודה המסכמת



Figure 5. Example validation images successfully classified by our method. For each image, the ground-truth label and the top-5 labels predicted by our method are listed.

- AlexNet (2012). דומה לLeNet שפותחה ב 1998.
- 18 שכבות חבויות. 16.4% טעות.
- שיפור משמעותי משנה קודמת שבה הרשת הטובה ביותר הגיעה ל 26% טעות.
- GPU, Rectifier, Dropout
- VGG(2014) 19 שכבות חבויות. 7.32%
- GoogleNet(2014) 22 שכבות חבויות. 6.67% טעות
- ResNet(2015) 152 שכבות חבויות. 4.94% טעות.
- **ממוצע הטעות של אנשים על ImageNet הוא 5.1% נקודה למחשבה האם אכן מכונות מאיימים על האנושות??**
- מודל מאוד אקספרסיבי, תוך שמירה על מספר פרמטרים כולל נמוך
- חלק מהנירונים במקום להתחבר לשכבה הסמוכה מדלגים על שכבות ומתחברים לנירונים בשכבות עמוקות יותר



## מסקנה מסקירת התחרות ומסקנות כלליות

---

1. לעומק הרשת יש חשיבות מכרעת!

2. Sparse hidden units activation אפקט מרכזי להפחתת התאמת יתר